免费提供
电子教案

高等院校规划教材
软件工程系列

构件式软件技术

王映辉　编著

机械工业出版社
CHINA MACHINE PRESS

高等院校规划教材·软件工程系列

构件式软件技术

王映辉　编著

机械工业出版社

本书从构件、服务、SOA 以及在网络环境下基于构件的软件开发通信机制与技术 4 个大的方面，本着原理讲解与实际开发相结合的理念，为读者呈现和阐述构件软件技术的详细内容。

本书可作为软件工程专业的本科课程教材，也可作为从事软件技术、信息技术等相关工作人员的参考书。

图书在版编目（CIP）数据

构件式软件技术/王映辉编著. —北京：机械工业出版社，2012.5
高等院校规划教材·软件工程系列
ISBN 978-7-111-37770-2

Ⅰ.①构… Ⅱ.①王… Ⅲ.①软件开发-高等学校-教材
Ⅳ.①TP311.52

中国版本图书馆 CIP 数据核字（2012）第 047413 号

机械工业出版社(北京市百万庄大街 22 号　　邮政编码 100037)
责任编辑：郝建伟　牟桂玲
责任印制：杨　曦
高等教育出版社印刷厂印刷

2012 年 5 月第 1 版·第 1 次印刷
184mm×260mm ·13 印张·318 千字
0001-3000 册
标准书号:ISBN 978-7-111-37770-2
定价:29.00 元

凡购本书,如有缺页、倒页、脱页,由本社发行部调换

电话服务
社服务中心:(010)88361066
销售一部:(010)68326294
销售二部:(010)88379649
读者购书热线:(010)88379203

网络服务
门户网:http://www.cmpbook.com
教材网:http://www.cmpedu.com

封面无防伪标均为盗版

前　言

从广义的角度看，软件的成分可以是任何有意义的并完成一定功能的软件元素，包括子过程、对象，甚至是一段有意义的文档等。但在实际的开发中，从目前的已成熟的实现技术来看，软件成分的表现形态可以是构件（Component）或者服务（Service）。本书将基于这些软件成分构造软件的技术称为构件式软件技术，这里的“构件式软件技术”中的“构件”就是指可被真正实现的构件和服务。

基于构件的软件设计与实现，已经成为软件开发的主流技术，也是一种全新的软件开发方法学的基本内容。为此，软件开发需要软件宏观蓝图作指导，这就是软件体系结构（Software Architecture）技术。软件体系结构随着软件的发展，特别是分布式网络环境的普及和应用，已进入到了 SOA（Service - Oriented Architecture，面向服务架构）的时代。服务作为软件在网络分布式环境下的另外一种软件成分形态，为网络软件的开发提供了更优良的支撑。

此外，在网络环境下架构软件和开发软件，软件成分之间的通信等相关问题就成为一个瓶颈问题。而微软的 WCF（Windows Communication Foundation，Windows 通信基础）为此提供了基本的支撑，也是在网络分布式环境下软件实现松散耦合的关键技术之一。

为此，本书的第 1 章简述了构件开发的环境工具，第 2 ~ 8 章对构件进行了详细讲解，第 9 和 11 章阐述了软件体系结构与 SOA，第 10 和 12 章介绍了服务和 WCF 及其在网络环境下对基于构件的软件开发支撑机制，第 13 章给出了一个具体的大型应用案例分析。

本书可作为软件工程专业的本科课程教材，也可作为从事软件技术、信息技术等相关工作人员的参考书。

最后，感谢本书所有参考文献的作者。可以说，没有这些参考文献，就没有本书的成稿和出版。此外，感谢陕西省科技厅（陕西省重大科技创新产业化项目：2009ZKC 02 - 08）和教育厅（产业化培育项目：09JC08）的支持，同时也感谢陈亚磊、孙洁、温晓燕、李爱民及本书的编辑给出的修改意见和辛勤劳动。

编　者

目　录

第 1 章　.NET 对现代软件开发的支持

Microsoft.NET（简称 .NET）是 Microsoft XML Web Services 的有效开发支撑平台。XML Web Services 允许应用程序通过 Internet 进行通信和共享数据，而与操作系统、设备或编程语言无关。Microsoft .NET 平台提供了创建 XML Web Services 并将这些服务集成在一起的环境和技术支撑。

1.1　.NET 概览

.NET 是一套将信息、各种系统和设备连接在一起并基于 Web 服务（Web Services）的软件技术，能有效支撑基于互联网平台的软件开发，是基于 XML 数据交换的 Web 服务平台。通过 Microsoft .NET 可以创建真正的分布 Web 服务，能集成这些服务并使这些服务协同工作，共同合作完成某个特定的任务。在此过程中 Web 服务还能够保持松耦合。

.NET 蕴含的基本理念是：不再关注单个的系统和与 Internet 连接的单个设备，而是要让所有的计算机群、相关设备和服务协同工作，提供涉及面更广、功能更全面的解决方案。用户和供应商可以将企业产品和服务无缝地嵌入在其自身的业务进程和信息系统中。

1.1.1　.NET 的组成

.NET 将互联网作为构建新一代操作系统的基础，并对互联网和操作系统的设计思想进行合理延伸，使开发人员能够创建出与设备无关的应用程序，以便轻松实现互联网连接。.NET 包括一个相当广泛的产品家族，它们构建于 XML（Extensible Markup Language，可扩展标记语言）和互联网产业标准之上，为用户提供 Web 服务的开发、管理、应用和体验。

1. .NET 开发平台

.NET 开发平台可用于建立 Web 服务应用程序和 Windows 桌面应用程序的软件构件，包括 .NET Framework（.NET 框架）、.NET 开发者工具和 ASP.NET，为开发新型的互动协作软件提供了一个先进的体系结构模型。

（1）.NET Framework

.NET Framework 主要由两部分组成：公共语言运行时（Common Language Runtime，CLR）和 .NET Framework 类库。

CLR 是 .NET Framework 的基础，可以将其看做一个在软件执行时管理代码的代理，它提供内存管理、线程管理和远程处理等核心服务，并且严格地实施类型安全。具体来说，.NET 开发平台是一组用于建立 Web 服务器应用程序和 Windows 桌面应用程序的“软件构件”集，用该平台创建的应用程序在 CLR 底层的控制下运行。由此可见，CLR 是一个软件引擎，用来加载应用程序，确认它们可以没有错误地执行，进行相应的安全许可验证，执行应用程序，然后在运行完成后将它们清除。

.NET Framework 类库提供了使应用程序可以读/写 XML 数据、在 Internet 上通信、访问

数据的代码。所有的类库都建立在一个基础的类库之上，它提供管理使用最为频繁的数据类型（如数值或文本字符串）的功能，以及诸如输入/输出等底层功能。具体来说，.NET Framework 类库是一个综合性的可复用类型的面向对象的集合，可以使用它开发多种应用程序，这些应用程序包括传统的命令行或 GUI（图形用户界面）应用程序，也包括基于 ASP.NET 所提供的最新概念下的应用程序（如 Web 窗体和 XML Web Services）。

（2）.NET 开发者工具

.NET 开发者工具包括：Visual Studio .NET 集成开发环境（Visual Studio .NET Integrated Development Environment），用来开发和测试应用程序；.NET 编程语言（如 Visual Basic .NET 和新的 Visual C#），用来创建运行在 CLR 下并且使用类库的应用程序。

（3）ASP.NET

ASP.NET 是一个取代以前的 ASP（Active Server Pages）的特殊类库，用来创建动态的 Web 内容和 Web 服务器应用程序，这些都将采用诸如 HTML、XML 和简单对象访问协议（Simple Object Access Protocol，SOAP）等 Internet 协议和数据格式。

Web 服务器应用程序通常依赖于 ASP.NET，ASP.NET 又依赖一个用于发送和接收 SOAP 信息的 Web Services 库，以及一个 Web 用户接口（UI，有时称为 Web 表单，是浏览器接收用户输入并动态生成的 Web 页面）。Windows 桌面应用程序通过使用 Windows 表单库（也称为 Windows 表单）可以显示一个图形 UI。

此外，Visual Studio .NET 还提供了一个用于在该平台上创建应用程序的图形集成开发环境。程序员可以使用一种或多种 .NET 编程语言来编写代码。这些语言包括微软自己的 Visual Basic .NET（VB.NET）、Visual C ++ 、Visual C#和 JScript .NET 等。

2. .NET 服务器

.NET 服务器是搭建 .NET 平台的后端基础。.NET 服务器包括：

1）Windows 2000 Server（包含 Advanced Server 和 Datacenter Server）。.NET 结构中，最重要的革新应属于 Web Services。Web Services 构架在 Web Server 之上，能够通过 SOAP 与用户端联系，并帮助用户端完成用户请求的服务。在 .NET Framework 中，Web Services 就是架构在 Windows 2000 Servers 支撑下的 IIS 5.0 之上的。

2）Exchange 2000 Server。Exchange 不是单纯的 E - mail Server，它更是一套不折不扣的信息交换平台。

3）SQL Server。SQL Server 提供完善数据处理功能，包含数据挖掘和 XML 的直接 Internet 支持。

4）BizTalk Server。BizTalk Server 用于企业间交换商务信息，它利用 XML 作为企业内部及企业间文档传输的数据格式，可以屏蔽平台、操作系统不同的差异，使商业系统的集成成为可能。

5）Commerce Server。Commerce Server 用于快速创建在线电子商务。

6）Mobile Information Server。Mobile Information Server 为移动解决方案提供可靠而且具有伸缩性的平台。

3．.NET 基础服务

.NET 基础服务是一套集成的 XML Web 服务，它以用户为中心，让用户来控制数据。.NET 基础服务包括护照（Passport，用于用户身份认证）和消息传送服务、文档储存、用户

首选参数管理，以及日历管理和其他功能。微软正在着力建设的 .NET My Services 等基础性服务平台，是为 .NET 基础构造的关键内容上提供一些基础服务，并使开发人员扩展这些基础服务。

4. .NET 终端设备

.NET 终端设备指提供 Internet 连接并实现 Web 服务的终端设备，如个人计算机、个人数字助理（PDA）设备及嵌入式设备等。

5 . .NET 用户体验

.NET 的最终目的就是让用户在任何时间、任何地方，利用任何设备都能访问所需的信息、文件和程序，并且用户不需要知道这些内容放在什么地方，只需要发出请求，然后等待接收即可，在此过程中所有后台的复杂性是完全被屏蔽的。最终用户与软件交互是通过新型的智能设备，并结合 XML Web 服务的集成来完成的，其体现的是个性化的服务，这就是 .NET 体系。从技术层面上来看，.NET 体验是 XML Web 服务在适当的时候与本地应用代码的一种结合。

微软正将以下 4 种普遍受欢迎的产品转换成 .NET 体验，见表 1-1。

表 1-1 .NET 体验

微软产品	通过 .NET 而增强的功能
Microsoft Office	传统的 Office 功能加上 Web 服务特色，如 SmartTags，实现重要数据信息的在线上下文下的连接
MSN	完全的客户体验，包括从各种不同的公司获取服务
MS Small Business Portal	针对中小型企业的体验，包括管理企业财务和库存及自动使用诸如 eBay 这样的销售机制服务等
MS Visual Studio 开发系统	通过将动态 MSDN（Microsoft Developer Net work，微软开发者网络）数据和共同的编码标准集成在一起，从而为开发人员提供一个丰富的开发环境

在上述 5 个组成部分当中，.NET 开发平台是 .NET 软件构造中最具挑战性的部分，其他 4 个部分则是紧紧围绕 .NET 框架来进行组织、整合的。

1.1.2 .NET 的特点

.NET 的特点如下：

1）.NET 是 Internet 上第一个大规模的高度分布式应用服务框架。

2）.NET 支持多种语言的互操作，即在一种语言环境下开发的构件，可在另一语言环境下得以复用。目前 . Net 支持的语言有 20 多种。

3）.NET 通过将各语言先编译成中间语言（Intermediate Language，IL），然后在执行时用即时（Just In Time）编译器将之编译成本地平台代码来实现异构平台下对象的互操作。目前 . Net 支持的平台有 Windows，而对 Linux 和 UNIX 的支持正在开发中。

4）.NET 通过 CLR 来确保对资源对象以及类型的安全。

5）.NET 通过对 HTTP、XML、SOAP 和 WSDL 等 Internet 标准的强劲支持，提供在异构网络环境下获取远程服务、连接远程设备和交互远程应用的编程界面。

6）.NET 使用了一个叫做“联盟”的管理程序，这个程序能全面管理平台中运行的服务

程序，并且为它们提供强大的安全后台保护。

1.2 .NET 对构件的支持

本节将从开发人员的需求和 .NET 平台自身的优势两方面，阐述 .NET 对构件的支持。

首先，.NET 满足开发人员的需求。我们知道构件不仅仅是可复用的程序代码片段，此外它还包括测试用例、设计文档、设计过程、需求分析文档，甚至领域知识等。这就决定了构件包括各式各样的文件类型，其中以程序代码开发出来的构件文件类型居多。对开发人员而言，最常见的构件就是程序代码片段，最常见文件类型有可执行文件（如 . exe 文件）、链接库（如 . dll 文件）、Applet（如 Java 中的 . class 文件）和 Web 页面（如 . htm 和 . html 文件）等。在 .NET 框架出现之前，如果要开发构件就需要安装多个编译应用程序的开发平台，不同的开发平台生成不同类型的构件文件，由于硬件水平的限制，因此在同一个计算机上安装多个开发平台实现起来很困难，这给开发人员带来很多不便。而 .NET 框架没有限制应用程序的类型，它可以创建 Windows 应用程序、Web 应用程序、Web 服务和其他各种类型的应用程序。这一点可以满足开发人员使用同一集成开发平台开发不同类型构件的需要。

其次，.NET 平台与其他开发平台相比，自身具有很强的优势。一方面，它可以作为集成各种操作系统的方式。尽管 .NET 框架的 Microsoft 版本仅运行在 Windows 操作系统上，但以后将推出运行在其他操作系统上的版本。另一方面，.NET 框架的设计方式保证它可以用于各种语言，包括 C#、C ++、Visual Basic 和 Jscript，甚至一些旧的语言，如 COBOL 等。为此，还推出了这些语言的 .NET 版本，目前还在不断推出更多的 .NET 版本的语言。所有这些语言都可以访问 .NET Framework，它们还可以彼此交互。例如，C#开发人员可以使用 Visual Basic 程序员编写的代码，反之亦然。

1.3 .NET 对服务的支持

1.3.1 .NET 对 SOA 的支持

面向服务架构（Service - Oriented Architecture，SOA）有时也称为面向服务的体系结构，是网络环境下备受业界关注的一个主题，它代表了软件架构的一种方向。SOA 是一种通过为所有软件提供服务外观（Service Facade），并将这些服务的 WSDL（Web Services Description Language，Web 服务描述语言）集中发布到一个地方的、将企业内软件组织到一起的方法。

SOA 服务由以下三部分组成：①将提供的服务类加以实现；② 支撑宿主服务的主机环境；③ 实现客户与一个或多个端点的连接。服务是 SOA 的基础，可以直接被使用，从而有效控制与软件代理交互的人为依赖性。

SOA 正处于发展中，也是目前 SOA 的一个重要特征，它代表一个开放的、敏捷的、可扩展的、可联邦的、可组合的架构，包含了自治的、高服务质量的、厂商多样性的、可互操作的、可发现的和潜在可复用的服务，并使用 Web 服务来实现。以下一些特征可由目前成熟的 .NET 技术来描述。

（1）基于开放标准

.NET 类库提供了一些包含支持工业标准、第一代 Web 服务规范类集的命名空间。

.NET 的 WSE（Web Services Enhancements，Web 服务改进）对特殊的“WS - * 规范集”提供了支持。值得关注的是，微软自身对几个关键性开放的 Web 服务规范的开发做出了重要贡献。

（2）支持厂商多样性

因为 ASP.NET Web 服务是遵循工业标准而创建的，所以在企业层面支持厂商的多样性，能够围绕一个 .NET SOA 来构建其他非 .NET SOA，并且只要 Web 服务遵循通用的标准就具有互操作性。

考虑到其开发或实现，.NET 框架提供有限的厂商多样性。这是因为它是一个属于单一厂商（微软）的专有技术。然而，由于第三方市场的存在，则可提供大量的增值产品。另外，一些服务器产品厂商支持部署，并支持 .NET Web 服务和程序集。

（3）内在互操作性

.NET Framework 2.0 与 Visual Studio 2005 一起，提供 WS - I［（Web Services Interoperability Organization，Web 服务互操作性组织）WS - I 是一种行业联盟，其章程是促进 Web 服务规范组织间的互操作性］基本体系的支持。这意味着使用 Visual Studio 2005 开发的 Web 服务默认是与基本体系兼容的，即前一版本的 Visual Studio 能用于开发与基本体系兼容的 Web 服务，但是它们需要使用第三方测试工具的支持来确保兼容性。

（4）促进联邦的形成

尽管 BizTalk 服务器平台技术不是 .NET 框架部分，但可将它视为一个使用扩展而达到跨越不同企业环境的联邦。它应用一系列的适配器，这些适配器可由第三方适配器市场来补充。BizTalk 还提供一个支持 BPEL（Business Process Execution Language，业务流程执行语言）的编排引擎，来完成流程定义的导入和导出。

（5）架构可组合性

.NET 类库是一个可组合编程模型的示例，因为类只提供功能。因此，这些功能只在 Web 服务实际需要时才导入，并在其底层的业务逻辑内被使用。

（6）可扩展性

ASP.NET Web 服务受制于设计标准及相关的最佳实践经验，它将从提供的可扩展服务接口及应用逻辑中获益（通过面向服务类设计的程序集而实现）。因此，可扩展性不是 .NET 框架的一个直接方法，而是用 .NET 技术的一个由用户发挥主动性的设计方法。此外，.NET SOA 通过兼容的平台产品也可实现功能扩展性，类似前面所介绍的 BizTalk 服务器。

（7）支持面向服务的建模

面向服务企业的终极状态是要获得业务与技术领域间的双丰收，这可以通过 SOA 获得，而 .NET 框架提供了一个 SOA 基础，能充分展示基本的 SOA 特征并呈现 SOA 的新品质。微软对 .NET 框架的扩展及第三方产品，为实现企业范围的松散耦合和建模提供了支撑。

（8）逻辑层抽象

.NET SOA 能将 ASP.NET Web 服务与服务层定位在不同层次上，并进行逻辑抽象。可将遗留的和新的应用逻辑封装，并全部通过适当的服务接口设计来抽象。可以使用定制的 SOAP 报头及相关的标识符或利用 WSE 扩展的优势构建服务组合，类似对 WS - 寻址（WS

- Addressing）和 WS - 推举（WS - Referral）所提供的支持。

因为 .NET 框架支持工业标准的 Web 服务开发，抽象层的创建可通过合适的定位与服务层的应用来实现。使用编排层能进一步增强对企业的响应，通过减轻特定业务流程逻辑，减少对以任务为核心服务的需要。

1.3.2 SOA 的 WCF 实现

企业架构师把 SOA 看做是一种工具，它可以帮助企业更快、更经济地响应变化的市场。WCF（Windows Communication Foundation，Windows 通信基础）提供了 SOA 技术基础，它支持 SOA 的设计和实现，提供了把资源与复用联系起来的能力。

Microsoft 的 WCF 的设计目标包括跨平台的互操作性、现有技术的融合和支持面向服务的开发。

随着 .NET 3.5 的发布，Microsoft 全面实现了上述目标。

为了最大化地实现跨平台的互操作性，WCF 的架构师选择 SOAP 作为原始的消息传递协议。这使得运行在 Windows 上的 WCF 应用程序有可能与遗留的应用程序、MacOS X、Linux、Windows 客户机、Solaris 以及遵守 WS - I 规范的其他任何计算机通信。

为了统一现有的技术，WCF 吸取了分布式系统技术所有特性，通过一个简单、清晰的 API（Application Programming Interface，应用程序编程接口），对于以前需要 ASMX、WSE、System. Messaging、.NET Remoting 及其他企业解决方案完成的事情，现在就可以轻松地做到，它们全都包含在 WCF 内。这有助于开发人员降低处理分布式技术的复杂性，将精力集中在实现上。

WCF 支持 SOA 的技术和实现，它允许在面向对象的基础上构建项目，并且具有分布式系统所需的面向服务的能力。

.NET 3.5 采用一种非常灵活的方法来编写程序，它提供了一组极佳的混合和匹配方式来处理开发者面对的大多数编程困难。开发者可以使用配置文件，通过编程方式深入对象模型，以及可以使用声明性编程。

WCF 从本质上讲是一个基于消息的系统，因此其各部分的功能都是通过消息来交换服务的。

1. 契约

WCF 契约（Contracts）十分类似于现实生活中签署的合同。例如，某人签署的一个合同可能包含工作类型信息和他想要提供给另一方的信息，WCF 契约包含了类似的信息。契约中的信息规定了服务要完成的工作以及服务提供的可用信息的类型。在 WCF 中，所有的服务者通过契约供外部进行调用。

契约主要有 4 种类型：数据契约、消息契约、服务契约和异常契约。契约及其使用方法的详细讲解可参阅相关资料，这里只介绍一些入门知识。

（1）数据契约

数据契约（Data Contract）明确规定服务将要交换的数据。服务和客户端不需要在数据类型上一致，但是必须在数据契约上一致，包括参数和返回类型。

（2）消息契约

消息契约（Message Contract）提供了比数据契约更多的实现控制的内容，因为它控制

的是服务发送和接收的 SOAP 消息。换句话说，消息契约允许定制 SOAP 消息中参数的类型格式。

（3）服务契约

服务契约（Service Contract）描述了客户端能调用服务的那些操作。可以把它想象成单个声明，这个声明只说明了“消息的数据类型、所处的位置以及在通信时使用的协议”。

（4）异常契约

异常契约（Fault Contract）描述了在服务中可能出现的异常以及服务如何处理、传播异常。

2. 服务运行时

在契约之下，WCF 的工作引擎就是服务运行时（Service Runtime），它指定并且管理着服务操作或运行库的行为（称为服务运行库行为）。服务行为控制服务类型行为，它们没有对端点或者消息行为的控制权。同样，端点和消息行为对服务行为也没有控制权。

服务运行时层管理的行为包括以下几种。

- 节流行为（Throttling Behavior）：决定被处理的数量。
- 错误行为（Error Behavior）：指定在服务运行期间出错时采取的动作。
- 元数据行为（Metadata Behavior）：控制元数据对外是否可见。
- 实例行为（Instance Behavior）：控制处理消息时可用服务实例的数量。
- 消息检查（Message Inspection）：使服务可以检查整个或部分消息。
- 事务行为（Transaction Behavior）：启动事务操作，即如果进程在服务运行期间失败，那么可以进行事务回滚。

3. 消息传送

消息传送（Messaging）层由通道组成。通道是以某种方式对消息进行处理（如对消息进行身份验证）的构件。一组通道也称为通道堆栈。通道对消息和消息头进行操作。这与服务运行时不同，服务运行时主要涉及对消息正文内容的处理。

通道主要有两种类型：传输通道和协议通道。

传输通道读取和写入来自网络（或外部的其他通信点）的消息。某些传输通道使用编码器来将消息（表示为 XML Infoset）转换为网络所使用的字节流的形式，或将字节流表示形式转换为消息。传输通道的示例包括 HTTP、命名管道、TCP 和 MSMQ（MicroSoft Message Queue，微软消息队列）。编码的示例包括 XML 和优化的二进制文件。

协议通道通过读取或写入消息头的方式来实现消息处理协议。此类协议的示例包括 WS - Security 和 WS - Reliability。消息传递层说明数据的可能格式和交换模式。WS - Security 是对在消息层启用安全性规范的实现。通过 WS - Reliability 可以保证消息的传递。编码器提供了大量的编码，可使用这些编码来满足消息的需要。HTTP 通道指定应使用超文本传输协议来传递消息。同理，TCP 通道指定 TCP。事务流通道控制已经过事务处理的消息模式。通过命名管道通道可以进行进程间通信。使用 MSMQ 通道可以与 MSMQ 应用程序进行互操作。

4. 宿主

WCF 服务可以运行在多种宿主（Hosting）之中。它可以运行在一个可执行文件或一个 Windows 服务中，这种运行方式称为自包含运行。还有一种运行方式是 WCF 服务在 IIS

或 WAS 服务中被自动激活来运行。这些运行时的行为和基础功能都是由宿主子系统负责的。

1.4 .NET 对软件框架的支持

1.4.1 .NET 框架简介

2000 年夏天，微软公司首次向公众发布 .NET 战略。大约 3 年后，微软公司以实际行动澄清 .NET 主要包含两部分内容：.NET 框架和 Visual Studio。其中，.NET 框架是由微软公司开发的一个致力于敏捷软件开发、快速应用开发，以及与平台无关性和网络透明化的软件开发平台；Visual Studio 是一种多语言集成开发环境，主要用于构建 .NET 框架应用程序。.NET 框架经历了如表 1-2 所示的不同版本。

表 1-2 .NET Framework 版本比较

版本	完整版本号	发行日期	Visual Studio	Windows 默认安装
1.0	1.0.3705.0	2002-2-13	Visual Studio .NET	
1.1	1.1.4322.573	2003-4-24	Visual Studio .NET 2003	Windows Server 2003
2.0	2.0.50727.42	2005-11-7	Visual Studio 2005	
3.0	3.0.4506.30	2006-11-6	Visual Studio 2005	Windows Vista, Windows Server 2008
3.5	3.5.21022.8	2007-11-19	Visual Studio 2008	Windows 7, Windows Server 2008 R2
4.0	4.0.30319.1	2010-4-12	Visual Studio 2010	

在 1.1.1 节已经介绍过，.NET 框架主要包括公共语言运行时和 .NET 框架类库两个构件。CLR 为构建应用程序提供了标准的基础设施，.NET 框架类库提供了一大套标准类，以任何语言编写的 .NET 框架应用程序都可以使用它们。

CLR 最早被称为下一代 Windows 服务运行时（NGWS Runtime）。它是直接建立在操作系统上的一个虚拟环境，主要的任务是管理代码的运行。CLR 现在支持几十种现代的编程语言，并以一种中间语言代码的形式被执行。同时，CLR 还提供了许多功能，以简化代码的开发和应用配置，同时也改善了应用程序的可靠性。如果某种语言的编译器是以运行时为目标的，那么利用该语言开发生成的代码在 .NET 中被称为托管代码。因为这样的代码是直接可以运行在 CLR 上的，所以具有与平台无关的特点。使用 .NET 框架编写的每一个应用程序都依赖于 CLR，他提供了一套通用数据类型，是 C#、VB 以及其他所有针对 .NET 框架而设计的语言基础。因为无论选择何种语言，都使用同一套基础设施，因此开发人员会看到一个更加协调一致的环境。

.NET 框架类库由一组广泛的、面向对象的、可被开发者用于任何编程语言的可重用类集合组成，可为应用程序提供各种高级的构件和服务。它将程序员从繁重的编程细节中解放出来，并专注于程序的商业逻辑，为开发各种应用程序提供支持，如支持传统的命令行程序、Windows 图形界面程序，或是面向 Internet 分布式计算平台的 ASP.NET 或 XML Web 服务的开发。在 .NET Framework 3.5 中新增了 WCF、WPF、WF 和 WCS 等框架类库，如图 1-1 所示。

.NET Framework 3.5 是一个成熟的框架，其底层类库依然调用的是 .NET 2.0 封装好的所

有类。.NET 3.5 支持目前最流行的开发语言 VB 和 C#。同时，.NET 3.5 自动包含 .NET 2.0 以及 .NET 3.0，并为这两个版本提供安全性修复的功能，同时也增加了少量的类库。此版本提供的新功能有：

- LINQ 支持，包括 LINQ to Object、LINQ to ADO.NET 以及 LINQ to XML。
- 利用 ASP.NET AJAX 可以创建更有效、更具交互性、高度个性化的 Web 页面。
- 用于生成 WCF 服务的全新 Web 协议支持，包括 AJAX、JSON（JavaScript Object Notation，是一种轻量级的数据交换格式）、REST（Representational State Transfer，表述性状态转移）、POX（Plain Old XML）、RSS（Really Simple Syndication，是站点用来和其他站点之间共享内容的简易方式）以及若干新的 WS-* 标准。

Visual Studio 2008 中提供了对 WF、WCF 和 WPF 的完整支持，其中包括支持工作流服务等这些新技术。

.NET 框架类库可以用于构建许多不同类型的应用程序。例如，以 ASP.NET 框架来构建的浏览器应用程序，可以使用 ADO.NET 框架来访问和存储数据。

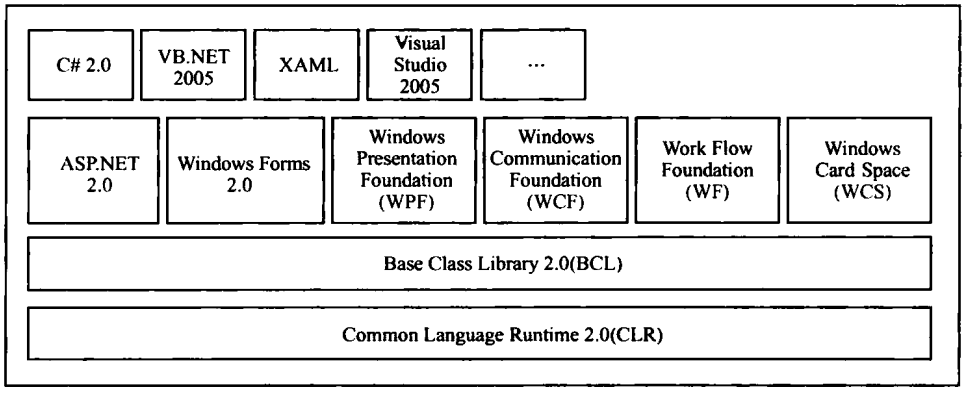

图 1-1　.NET 3.5 构件关系图

1.4.2　.NET 框架类库

我们将框架形象地比喻为一个骨架，它封装了某领域内处理流程的控制逻辑，所以我们经常说框架是一个半成品的应用程序。由于领域的种类众多，没有一个万能的框架可以应用于所有的领域。框架的目标性非常强，它专注于解决某一特定领域的问题，并致力于为这一特定领域提供通用的解决方案，所以框架必须具有针对性。例如，专门用于解决底层通信的框架，或专门用于医疗领域的框架。框架中也包含了很多元素，但是这些元素之间关系的紧密程度要远远大于类库中元素之间的关系。框架中的所有元素都为了实现一个共同的目标而相互协作。框架可以让我们的流程更清晰，代码更规范。图 1-2 所示的是 .NET 3.5 的简化框架。

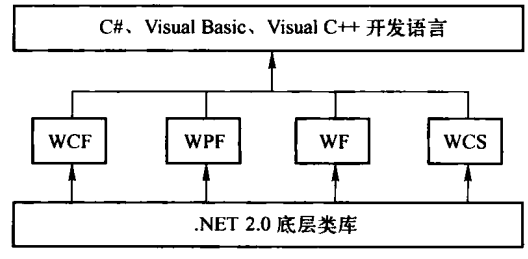

图 1-2　.NET 3.5 框架的基本组成

下面对 .NET Framework 3.5 中的相关内容进行描述。